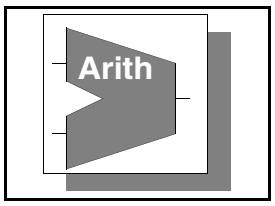
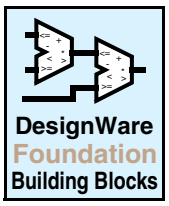




DW01_csa

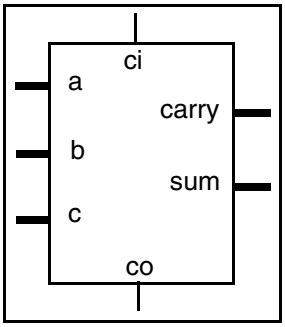
Carry Save Adder

Version, STAR, and myDesignWare Subscriptions: IP Directory

Features and Benefits

- Parameterized word length
- Carry-in and carry-out signals

Revision History



Description

DW01_csa adds three operands `a`, `b`, and `c` with a carry-in (`ci`) to produce the outputs sum and carry with a carry-out (`co`).

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
a	*width* bits	Input	Input data
b	*width* bits	Input	Input data
c	*width* bits	Input	Input data
ci	1 bit	Input	Carry-in
carry	*width* bits	Output	Carry output data
sum	*width* bits	Output	Sum output data
co	1 bit	Output	Carry-out

Table 1-2 Parameter Description

Parameter	Values	Description
width	≥ 1	Word length of `a`, `b`, `c`, `sum`, and `carry`

Table 1-3 Synthesis Implementations

Implementation Name	Function	License Feature Required
str	Synthesis model	DesignWare

Table 1-4 Simulation Models

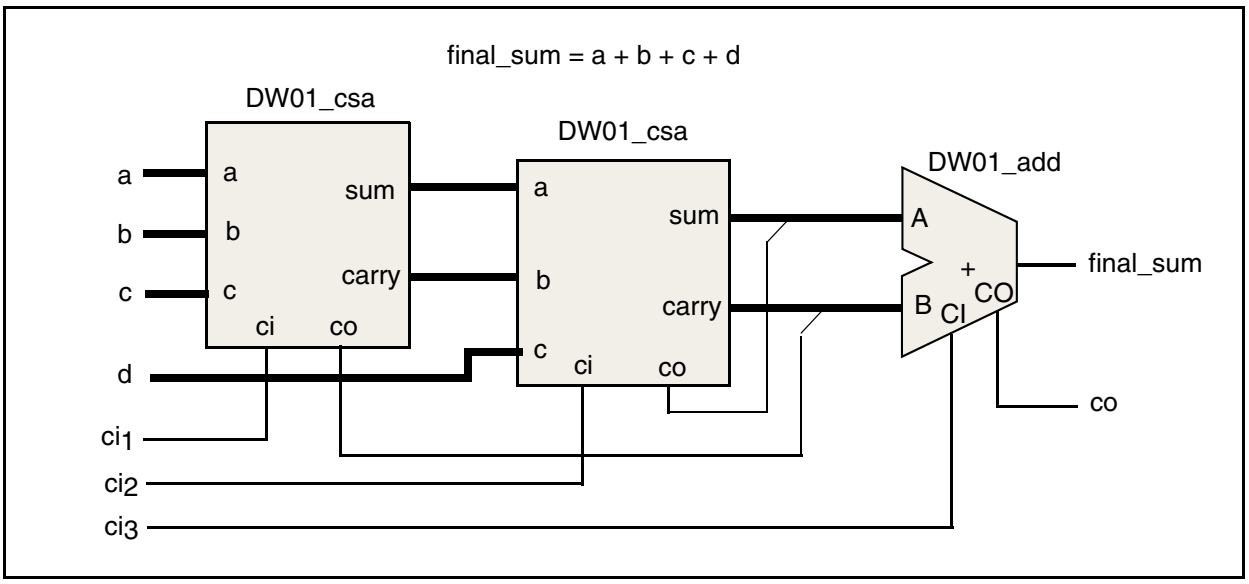
Model	Function
DW01.DW01_CSA_CFG_SIM	Design unit name for VHDL simulation
dw/dw01/src/DW01_csa_sim.vhd	VHDL simulation model source code
dw/dw01/src_ver/DW01_csa.v	Verilog simulation model source code

The addition is done without carry propagation, resulting in fast, constant-time operation and yielding a result in redundant carry-save representation. The `sum` and `carry` outputs can be added together to form the final (non-redundant) sum or they can be fed into another instance of DW01_csa to form a hierarchical summation tree.

Note that coding an HDL operator for K n-bit addition can be a superior alternative to using an instance of the DW01_csa component.

Figure 1-1 shows an example application that uses two instances of DW01_csa and one instance of DW01_add to compute the sum `a + b + c + d` with only one carry propagation, which is faster and smaller than using three instances of DW01_add.

Figure 1-1 Example Application



Related Topics

- Math – Arithmetic Overview
- DesignWare Building Block IP User Guide

HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DWpackages.all;
use DWARE.DW_foundation_comp.all;

entity DW01_csa_inst is
  generic ( inst_width : INTEGER := 8 );
  port ( inst_a     : in std_logic_vector(inst_width-1 downto 0);
         inst_b     : in std_logic_vector(inst_width-1 downto 0);
         inst_c     : in std_logic_vector(inst_width-1 downto 0);
         inst_ci    : in std_logic;
         carry_inst : out std_logic_vector(inst_width-1 downto 0);
         sum_inst   : out std_logic_vector(inst_width-1 downto 0);
         co_inst    : out std_logic );
end DW01_csa_inst;

architecture inst of DW01_csa_inst is
begin

  -- Instance of DW01_csa
  U1 : DW01_csa
    generic map ( width => inst_width )
    port map ( a => inst_a, b => inst_b, c => inst_c, ci => inst_ci,
               carry => carry_inst, sum => sum_inst, co => co_inst );
end inst;

-- pragma translate_off
configuration DW01_csa_inst_cfg_inst of DW01_csa_inst is
  for inst
  end for; -- inst
end DW01_csa_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
module DW01_csa_inst( inst_a, inst_b, inst_c, inst_ci, carry_inst,
                      sum_inst, co_inst );

  parameter width = 8;

  input [width-1 : 0] inst_a;
  input [width-1 : 0] inst_b;
  input [width-1 : 0] inst_c;
  input inst_ci;
  output [width-1 : 0] carry_inst;
  output [width-1 : 0] sum_inst;
  output co_inst;

  // Instance of DW01_csa
  DW01_csa #(width)
    U1 ( .a(inst_a), .b(inst_b), .c(inst_c), .ci(inst_ci),
         .carry(carry_inst), .sum(sum_inst), .co(co_inst) );

endmodule
```

Revision History

For notes about this release, see the *DesignWare Building Block IP Release Notes*.

For lists of both known and fixed issues for this component, refer to the STAR report.

For a version of this datasheet with visible change bars, click here.

Date	Release	Updates
June 2022	DWBB_202203.2	■ Above Figure 1-1 on page 2, clarified that coding an HDL operator for K n-bit addition can be a superior alternative to instantiating the DW01_csa component
January 2019	DWBB_201806.5	■ Updated example in "HDL Usage Through Component Instantiation - VHDL" on page 3 ■ Added this Revision History table and the document links on this page

Copyright Notice and Proprietary Information

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
www.synopsys.com